File: edt-user.doc --- EDT Emulation User Instructions

For GNU Emacs 19

Author: Kevin Gallagher <[redacted]>
Maintainer: Kevin Gallagher <[redacted]>
Keywords: emulations

==========

I. OVERVIEW:

This is Version 4.0 of the EDT Emulation for Emacs 19 and above.
It comes with special functions which replicate nearly all of EDT's
keypad mode behavior. It sets up default keypad and function key
bindings which closely match those found in EDT. Support is
provided so that users may reconfigure most keypad and function key
bindings to their own liking.

NOTE: Version 4.0 contains several enhancements. See the
Enhancements section below for the details.

Quick Start:

To start the EDT Emulation, first start Emacs and then enter

   M-x edt-emulation-on

to begin the emulation. After initialization is complete, the
following message will appear below the status line informing you
that the emulation has been enabled: "Default EDT keymap active".

   You can have the EDT Emulation start up automatically, each time
you initiate a GNU Emacs session, by adding the following line to
your .emacs file:

   (add-hook term-setup-hook 'edt-emulation-on)

IMPORTANT: Be sure to read the rest of this file. It contains very
           useful information on how the EDT Emulation behaves and how
           to customize it to your liking.

The EDT emulation consists of the following files:

edt-user.doc   - User Instructions and Sample Customization File
edt.el   - EDT Emulation Functions and Default Configuration
edt-lk201.el   - Built-in support for DEC LK-201 Keyboards
edt-vt100.el   - Built-in support for DEC VT-100 (and above) terminals
edt-pc.el   - Built-in support for PC 101 Keyboards under MS-DOS
edt-mapper.el   - Create an EDT LK-201 Map File for Keyboards Without
                       Built-in Support

Enhancements:

Version 4.0 contains the following enhancements:

 1. Scroll margins at the top and bottom of the window are now
     supported. (The design was copied from tpu-extras.el.) By
     default, this feature is enabled with the top margin set to
     10% of the window and the bottom margin set to 15% of the
     window. To change these settings, you can invoke the function
     edt-set-scroll-margins in your .emacs file. For example, the
     following line

          (edt-set-scroll-margins "20%" "25%")

     sets the top margin to 20% of the window and the bottom margin
     to 25% of the window. To disable this feature, set each
     margin to 0%. You can also invoke edt-set-scroll-margins
     interactively while EDT Emulation is active to change the
     settings for that session.

NOTE: Another way to set the scroll margins is to use the Emacs customization feature (not available in Emacs 19) to set the following two variables directly:

edt-top-scroll-margin and edt-bottom-scroll-margin

Enter the Emacs 'customize' command. First select the Editing group and then select the Emulations group. Finally, select the Edt group and follow the directions.

2. The SUBS command is now supported and bound to GOLD-Enter by default. (This design was copied from tpu-edt.el.) Note, in earlier versions of EDT Emulation, GOLD-Enter was assigned to the Emacs function 'query-replace'. The binding of 'query-replace' has been moved to GOLD-/. If you prefer to restore 'query-replace' to GOLD-Enter, then use an EDT user customization file, edt-user.el, to do this. See edt-user.doc for details.

3. EDT Emulation now also works in XEmacs, including the highlighting of selected text.

4. If you access a workstation using an X Server, observe that the initialization file generated by edt-mapper.el will now contain the name of the X Server vendor. This is a convenience for those who have access to their Unix account from more than one type of X Server. Since different X Servers typically require different EDT emulation initialization files, edt-mapper.el will now generate these different initialization files and save them with different names. Then, the correct initialization file for the particular X server in use is loaded correctly automatically.

5. Also, edt-mapper.el is now capable of binding an ASCII key sequence, providing the ASCII key sequence prefix is already known by Emacs to be a prefix. As a result of providing this support, some terminal/keyboard/window system configurations, which don't have a complete set of sensible function key bindings built into Emacs in 'function-key-map', can still be configured for use with EDT Emulation. (Note: In a few rare circumstances this does not work properly. In particular, it does not work if a subset of the leading ASCII characters in a key sequence are recognized by Emacs as having an existing binding. For example, if the keypad 7 (KP-7) key generates the sequence \"<ESC>Ow\" and \"<ESC>O\" is already bound to a function, pressing KP-7 when told to do so by edt-mapper.el will result in edt-mapper.el incorrectly mapping \"<ESC>O\" to KP-7 and \"w\" to KP-8. If something like this happens to you, it is probably a bug in the support for your keyboard within Emacs OR a bug in the Unix termcap/terminfo support for your terminal OR a bug in the terminal emulation software you are using.)

6. The edt-quit function (bound to GOLD-q by default) has been modified to warn the user when file-related buffer modifications exist. It now cautions the user that those modifications will be lost if the user quits without saving those buffers.

Goals:

1. Emulate EDT Keypad Mode commands closely so that current EDT users

will find that it easy and comfortable to use GNU Emacs with a small learning curve;

2. Make it easy for a user to customize EDT emulation key bindings without knowing much about Emacs Lisp;

3. Make it easy to switch between the original EDT default bindings and the user's customized EDT bindings, without having to exit Emacs.

4. Provide support for some TPU/EVE functions not supported in EDT.

5. Provide an easy way to restore ALL original Emacs key bindings, just as they existed before the EDT emulation was first invoked.

6. Support GNU Emacs 19 and higher. (GNU Emacs 18 and below is no longer supported.) XEmacs 19, and above, is also supported.

7. Supports highlighting of marked text within the EDT emulation on all platforms on which Emacs supports highlighting of marked text.

8. Handle terminal configuration interactively for most terminal configurations, when the emulation is invoked for the first time.

9. Support a PC AT keyboard under MS-DOS.

II. TERMINALS/KEYBOARDS SUPPORTED:

Keyboards used under a Window System are supported via the edt-mapper function. The first time you invoke the emulation under a window system, the edt-mapper function is run automatically and the user is prompted to identify which keys the emulation is to use for the standard keypad and function keys EDT expects (e.g., PF1, PF2, KP0, KP1, F1, F2, etc.). This configuration is saved to disk read each time the emulation is invoked.

In character oriented connections not running a window manager, built-in support for the following terminals/keyboards is provided:

(1) DEC VT-100 series and higher. This includes well behaved VT clones and emulators. If you are using a VT series terminal, be sure that the term environment variable is set properly before invoking emacs.

(2) PC AT keyboard under MS-DOS.

Be sure to read the SPECIAL NOTES FOR SOME PLATFORMS sections to see if those notes apply to you.

III. STARTING THE EDT EMULATION:

Start up GNU Emacs and enter "M-x edt-emulation-on" to begin the emulation. After initialization is complete, the following message will appear below the status line informing you that the emulation has been enabled:

Default EDT keymap active

You can have the EDT Emulation start up automatically, each time you initiate a GNU Emacs session, by adding the following line to your .emacs file:

(add-hook term-setup-hook 'edt-emulation-on)

A reference sheet is included (later on) listing the default EDT Emulation key bindings. This sheet is also accessible on line from within Emacs by pressing PF2, GOLD H, or HELP (when in the EDT Default Mode).

It is easy to customize key bindings in the EDT Emulation. (See CUSTOMIZING section, below.) Customizations are placed in a file called edt-user.el. (A sample edt-user.el file can be found in the CUSTOMIZING section.) If edt-user.el is found in your GNU Emacs load path during EDT Emulation initialization, then the following message will appear below the status line indicating that the emulation has been enabled, enhanced by your own customizations:

User EDT custom keymap active

Once enabled, it is easy to switch back and forth between your customized EDT Emulation key bindings and the default EDT Emulation key bindings. (See the sample edt-user.el file below. Look at the binding to GOLD Z.) It is also easy to turn off the emulation (via the command edt-emulation-off). Doing so completely restores the original key bindings in effect just prior to invoking the emulation.

Emacs binds keys to ASCII control characters and so does the real EDT. Where EDT key bindings and GNU Emacs key bindings conflict, the default GNU Emacs key bindings are retained by the EDT emulation by default. If you are a diehard EDT user you may not like this. The CUSTOMIZING section explains how to change this so that the EDT bindings to ASCII control characters override the default Emacs bindings.

IV. SPECIAL NOTES FOR SOME PLATFORMS:

Sun Workstations running X:

Some earlier Sun keyboards do not have arrow keys separate from the keypad keys. It is difficult to emulate the full EDT keypad and still retain use of the arrow keys on such keyboards.

The Sun Type 5 and other more recent Sun keyboards, however, do have separate arrow keys. This makes them candidates for setting up a reasonable EDT keypad emulation.

Depending upon the configuration of the version of X installed on your system, you may find the default X keynames for the keypad keys don't permit Emacs to interpret some or all the keypad keys as something other than arrow keys, numeric keys, Home, PgUP, etc. Both Sun and HP have been particularly guilty of making bizarre keysym assignments to the keypad keys.

In most cases, the X Windows command, xmodmap, can be used to correct the problem. Here's a sample .xmodmaprc file which corrects this problem on one Sun workstation configuration using an older SunOS release configured with a Sun Type 5 keyboard:

```
! File:  .xmodmaprc
!
! Set up Sun Type 5 keypad for use with the GNU Emacs EDT Emulation
!
keycode  53 = KP_Divide
keycode  54 = KP_Multiply
keycode  57 = KP_Decimal
keycode  75 = KP_7
keycode  76 = KP_8
keycode  77 = KP_9
keycode  78 = KP_Subtract
keycode  97 = KP_Enter
keycode  98 = KP_4
keycode  99 = KP_5
keycode 100 = KP_6
keycode 101 = KP_0
keycode 105 = F24
keycode 119 = KP_1
keycode 120 = KP_2
keycode 121 = KP_3
keycode 132 = KP_Add
```

If edt-mapper.el does not recognize your keypad keys as unique keys, use the command

```
xmodmap -pke
```

to get a listing of the actual key codes and the keysyms mapped to them and then generate you own custom .xmodmaprc similar to the one above.

Next, feed .xmodmaprc to the xmodmap command and all the Sun Type 5 keypad keys will now be configurable for the emulation of an LK-201 keypad (less the comma key). In this example, the line

```
keycode 105 = F24
```

changes the X Windows name of the keypad NumLock key to be known internally as the F24 key. Doing so permits it to be configured to behave as the PF1 (Gold) key.

The side effect of this change is that you will no longer have a NumLock key. If you are using other software under X which requires a NumLock key, then examine your keyboard and look for one you don't use and redefine it to be the NumLock key. Basically, you need to clear the NumLock key from being assigned as a modifier, assign it to the key of your choice, and then add it back as a modifier. (See the "General Notes on Using NumLock for the PF1 Key on a Unix System" section below for further help on how to do this.)

PC users running MS-DOS:

By default, F1 is configured to emulate the PF1 (GOLD) key. But NumLock can be used instead if you load a freeware TSR distributed with MS-Kermit, call [redacted] This was once distributed in a file called gold22.zip and came with the source code as well as a loadable binary image. (See edt-pc.el in the Emacs lisp/emulation directory for more information.)

PC users running GNU/Linux:

The default X server configuration varies from distribution to distribution and release to release of GNU/Linux. If your system fails to recognize the keypad keys as distinct keys, change the NumLock state, turning it on or off, as the case may be, then try again. If this doesn't solve your problem, you may have to modify the X keysym mappings with xmodmap.

On one distribution on an Intel PC, the following .xmodmaprc set things up nicely.

```
! File:  .xmodmaprc
!
! Set up PC keypad under GNU/Linux for the GNU Emacs EDT Emulation
!
clear  mod2
keycode  77 = F12
keycode  96 = Num_Lock Pointer_EnableKeys
add mod2 = Num_Lock
```

In this example, after feeding the file to the xmodmap command, the PC NumLock keypad key will be configurable for the emulation of the PF1 key. The PC keypad can now emulate an LK-201 keypad (less the comma key), the standard keyboard supplied with DEC terminals VT-200 and above. This .xmodmaprc file switches the role of the F12 and NumLock keys. It has been tested on RedHat GNU/Linux 5.2. Other versions of GNU/Linux may require different keycodes. (See the "General Notes on Using NumLock for the PF1 Key on a Unix System" section below for further help on how to do this.)

NOTE: Remember, it may be necessary to have NumLock in one position (ON) or the other (OFF) for the PC keypad to emulate the LK-201 keypad properly.

General Notes on Using NumLock for the PF1 Key on a Unix System:

Making the physical NumLock key available for use in the EDT Emulation requires some modification to the default X Window settings. Since the keycode assignments vary from system to system, some investigation is needed to see how to do this on a particular system.

You will need to look at the output generated by xmodmap invoked with the "-pm" switch. examined. For example, on RedHat GNU/Linux 5.2 on a PC, we get the following output when running xmodmap.

"xmodmap -pm" yields:

```
xmodmap:  up to 2 keys per modifier, (keycodes in parentheses):

shift       Shift_L (0x32),  Shift_R (0x3e)
lock        Caps_Lock (0x42)
control     Control_L (0x25),  Control_R (0x6d)
mod1        Alt_L (0x40),  Alt_R (0x71)
mod2        Num_Lock (0x4d)
mod3
mod4
mod5        Scroll_Lock (0x4e)
```

Note that Num_Lock is assigned to the modifier mod2. This is what hides Num_Lock from being seen by Emacs.

Now, "xmodmap -pke" yields:

```
        .
        .
        .
keycode  77 = Num_Lock Pointer_EnableKeys
        .
        .
        .
keycode  96 = F12
        .
        .
        .
```

So, in RedHat GNU/Linux 5.2 on a PC, Num_Lock generates keycode 77. The following steps are taken:

1. clear the assignment of Num_Lock to mod2;
2. swap the keycodes assigned to F12 and Num_Lock;
3. assign Num_Lock back to mod2.

The .xmodmaprc file looks like this:

```
! File:  .xmodmaprc
!
! Set up PC keypad under GNU/Linux for the GNU Emacs EDT Emulation
!
clear  mod2
keycode  77 = F12
keycode  96 = Num_Lock Pointer_EnableKeys
add mod2 = Num_Lock
```

So, after executing "xmodmap .xmodmaprc", a press of the physical F12 key looks like a Num_Lock keypress to X. Also, a press of the

physical NumLock key looks like a press of the F12 key to X.

Now, edt-mapper.el will see "f12" when the physical NumLock key is pressed, allowing the NumLock key to be used as the EDT PF1 (Gold) key.

V. HOW DOES THIS EDT EMULATION DIFFER FROM REAL EDT?:

In general, you will find that this emulation of EDT replicates most, but not all, of EDT's most used Keypad Mode editing functions and behavior. It is not perfect, but most EDT users who have tried the emulation agree that it is quite good enough to make it easy for die-hard EDT users to move over to using GNU Emacs.

Here's a list of the most important differences between EDT and this GNU Emacs EDT Emulation. The list is short but you must be aware of these differences if you are to use the EDT Emulation effectively.

1. Entering repeat counts works a little differently than in EDT.

   EDT allows users to enter a repeat count before entering a command that accepts repeat counts. For example, when using the real EDT, pressing these three keys in sequence, GOLD 5 KP1, will move the cursor in the current direction 5 words. This does NOT work in Emacs!

   Emacs provides two ways to enter repeat counts and neither involves using the GOLD key. First, repeat counts can be entered in Emacs by using the ESC key. For example, pressing these keys in sequence, ESC 1 0 KP1, will move the cursor in the current direction 10 words. Second, Emacs provides another command called universal-argument that can be used to do the same thing. Normally, in Emacs has this bound to C-u.

2. EDT's line mode commands and nokeypad mode commands are NOT supported (with one important exception; see item 8 in the Highlights section below). Although, at first, this may seem like a big omission, the set of built-in Emacs commands provides a much richer set of capabilities which more than make up for this omission.

   To enter Emacs commands not bound to keys, you can press GOLD KP7 or the DO key. Emacs will display its own command prompt "M-x". This stands for the keypress Meta-x, where Meta is a special shift key. The Alt key is often mapped to behave as a Meta key. So, you can also invoke this prompt by pressing Meta-x. Typing the sequence "ESC x" will also invoke the prompt.

3. Selected text is highlighted ONLY on systems where Emacs supports the highlighting of text.

4. Just like in TPU/EVE, the ENTER key is NOT used to terminate input when the editor prompts you for input. The RETURN key is used, instead. (KP4 and KP5 (the direction keys) do terminate input for the FIND command, just like in EDT, however.)

VI. SOME HIGHLIGHTS IN THIS EDT EMULATION, AND SOME COMPARISONS TO THE ORIGINAL GNU EMACS EDT EMULATION:

1. The EDT define key command is supported (edt-define-key) and is bound to C-k in the default EDT mode when EDT control sequence bindings are enabled or one of the sample edt-user.el customization files is used. The TPU/EVE learn command is supported but not bound to a key in the default EDT mode but is bound in the sample edt-user.el file.

   Unlike the TPU/EVE learn command, which uses one key to begin the learn sequence, C-l, and another command to remember the sequence, C-r, this version of the learn command (edt-learn) serves as a toggle to both begin and to remember the learn sequence.

   Many users who change the meaning of a key with the define key and the learn commands, would like to be able to restore the original key binding without having to quit and restart emacs. So a restore key command is provided to do just that. When invoked, it prompts you to press the key to which you wish the last replaced key definition restored. It is bound to GOLD C-k in the default EDT mode when EDT control sequence bindings are enabled or one of the sample edt-user.el customization files is used.

2. Direction support is fully supported.

3. All original Emacs bindings are fully restored when EDT emulation is turned off. So, if a fellow worker comes over to your terminal to help you with a software problem, for example, and is completely confused by your EDT emulation bindings, just enter the command, edt-emulation-off, at the M-x prompt and the original Emacs bindings will be restored. To resume the EDT emulation, just enter edt-emulation-on.

4. User custom EDT bindings are kept separate from the default EDT bindings. One can toggle back and forth between the custom EDT bindings and default EDT bindings.

5. The Emacs functions in edt.el attempt to emulate, where practical, the exact behavior of the corresponding EDT keypad mode commands. In a few cases, the emulation is not exact, but we hope you will agree it is close enough. In a very few cases, we chose to use the Emacs way of handling things. As mentioned earlier, we do not emulate the EDT SUBS command. Instead, we chose to use the Emacs query-replace function, which we find to be easier to use.

6. Emacs uses the regexp assigned to page-delimiter to determine what marks a page break. This is normally "^\f", which causes the edt-page command to ignore form feeds not located at the beginning of a line. To emulate the EDT PAGE command exactly, page-delimiter is set to "\f" when EDT emulation is turned on, and restored to "^\f" when EDT emulation is turned off. But, since some users prefer the Emacs definition of a page break, or may wish to preserve a customized definition of page break, one can override the EDT definition by placing

   ```
   (setq edt-keep-current-page-delimiter t)
   ```

   in your .emacs file. Or, you can used the Emacs customize command to change its setting.

7. The EDT definition of a section of a terminal window is hardwired to be 16 lines of its one-and-only 24-line window (the EDT SECT command bound to KP8). That's two-thirds of the window at a time. Since Emacs, like TPU/EVE, can handle multiple windows of sizes of other than 24 lines, the definition of section used here has been modified to two-thirds of the

current window. (There is also an edt-scroll-window function which you may prefer over the SECT emulation.)

8. Cursor movement and deletion involving word entities is identical to EDT. This, above all else, gives the die-hard EDT user a sense of being at home. Also, an emulation of EDT's SET ENTITY WORD command is provided, for those users who like to customize movement by a word at a time to their own liking.

9. EDT's FIND and FNDNXT are supported.

10. EDT's APPEND, REPLACE, and SUBS commands are supported.

11. CHNGCASE is supported. It works on individual characters or selected text, if SELECT is active. In addition, two new commands are provided: edt-lowercase and edt-uppercase. They work on individual WORDS or selected text, if SELECT is active.

12. Form feed and tab insert commands are supported.

13. A new command, edt-duplicate-word, is provided. If you experiment with it, you might find it to be surprisingly useful and may wonder how you ever got along without it! It is assigned to C-j in the sample edt-user.el customization files.

14. TPU/EVE's Rectangular Cut and Paste functions (originally from the EVE-Plus package) are supported. But unlike the TPU/EVE versions, these here support both insert and overwrite modes. The seven rectangular functions are bound to F7, F8, GOLD-F8, F9, GOLD-F9, F10, and GOLD-F10 in the default EDT mode.

15. The original EDT emulation package set up many default regular and GOLD bindings. We tried to preserve most (but not all!) of these, so users of the original emulation package will feel more at home.

    Nevertheless, there are still many GOLD key sequences which are not bound to any functions. These are prime candidates to use for your own customizations.

    Also, there are several commands in edt.el not bound to any key. So, you will find it worthwhile to look through edt.el for functions you may wish to add to your personal customized bindings.

16. The VT200/VT300 series terminals steal the function keys F1 to F5 for their own use. These do not generate signals which are sent to the host. So, edt.el does not assign any default bindings to F1 through F5.

    In addition, our VT220 terminals generate an interrupt when the F6 key is pressed (^C or ^Y, can't remember which) and not the character sequence documented in the manual. So, binding emacs commands to F6 will not work if your terminal behaves the same way.

17. The VT220 terminal has no ESC, BS, nor LF keys, as does a VT100. So the default EDT bindings adopt the standard DEC convention of having the F11, F12, and F13 keys, on a VT200 series (and above) terminal, assigned to the same EDT functions that are bound to ESC, BS, and LF on a VT100 terminal.

18. Each user, through the use of a private edt-user.el file, can customize, very easily, personal EDT emulation bindings.

19. The EDT SELECT and RESET functions are supported. However, unlike EDT, pressing RESET to cancel text selection does NOT reset the existing setting of the current direction.

We also provide a TPU/EVE like version of the single SELECT/RESET function, called edt-toggle-select, which makes the EDT SELECT function into a toggle on/off switch. That is, if selection is ON, pressing SELECT again turns selection off (cancels selection). This function is used in the sample edt-user.el customization files.

20. EDT scroll margins are supported, but are disabled by default. (See CUSTOMIZING section below for instructions on how to enable them.)

VII. CUSTOMIZING:

Most EDT users, at one time or another, make some custom key bindings, or use someone else's custom key bindings, which they come to depend upon just as if they were built-in bindings. This EDT Emulation for GNU Emacs is designed to make it easy to customize bindings.

If you wish to customize the EDT Emulation to use some of your own key bindings, you need to make a private version of edt-user.el in your own private lisp directory. There are two sample files edt-user.el1 and edt-user.el2 for you to use as templates and for ideas. Look at edt-user.el1 first. Unless you will be using two or more very different types of terminals on the same system, you need not look at edt-user.el2.

First, you need to have your own private lisp directory, say ~/lisp, and you should add it to the GNU Emacs load path.

NOTE: A few sites have different load-path requirements, so the above directions may need some modification if your site has such special needs.

Creating your own edt-user.el file:

A sample edt-user.el file is attached to the end of this user documentation. You should use it as a guide to learn how you can customize EDT emulation bindings to your own liking. Names used to identify the set of LK-201 keypad and function keys are:

Keypad Keys:

```
                PF1 PF2 PF3 PF4
                KP7 KP8 KP9 KP-
                KP4 KP5 KP6 KP,
                KP1 KP2 KP3
                KP0     KPP KPE
```

Arrow Keys:

```
                LEFT RIGHT DOWN UP
```

Function Keys:

```
                F1 F2 F3 F4 F5  F6 F7 F8 F9 F10  F11 F12 F13 F14
                HELP DO  F17 F18 F19 F20

                FIND   INSERT   REMOVE
                SELECT PREVIOUS NEXT
```

Note:

Many VT-200 terminals, and above, steal function keys F1 thru F5 for terminal setup control and don't send anything to the host if pressed. So customizing bindings to these keys may not work for you.

There are three basic functions that do the EDT emulation custom bindings: edt-bind-key, edt-bind-gold-key, and edt-bind-function-key.

The first two are for binding functions to keys which are standard across most keyboards. This makes them keyboard independent, making it possible to define these key bindings for all terminals in the file edt.el.

The first, edt-bind-key, is used typically to bind emacs commands to control keys, although some people use it to bind commands to other keys, as well. (For example, some people use it to bind the VT200 seldom used

back-tick key (`) to the function "ESC-prefix" so it will behave like an ESC key.) The second function, edt-bind-gold-key, is used to bind emacs commands to gold key sequences involving alpha-numeric keys, special character keys, and control keys.

The third function, edt-bind-function-key, is terminal dependent and is defined in a terminal specific file (see edt-vt100.el for example). It is used to bind emacs commands to LK-201 function keys, to keypad keys, and to gold sequences of those keys.

SPECIFYING WORD ENTITIES:

The variable edt-word-entities is used to emulate EDT's SET ENTITY WORD command. It contains a list of characters to be treated as words in themselves. If the user does not define edt-word-entities in his/her .emacs file, then it is set up with the EDT default containing only TAB.

The characters are stored in the list by their numerical values, not as strings. Emacs supports several ways to specify the numerical value of a character. One method is to use the question mark: ?A means the numerical value for A, ?/ means the numerical value for /, and so on. Several unprintable characters have special representations:

```
?\b  specifies  BS, C-h
?\t  specifies  TAB, C-i
?\n  specifies  LFD, C-j
?\v  specifies  VTAB, C-k
?\f  specifies  FF, C-l
?\r  specifies  CR, C-m
?\e  specifies  ESC, C-[
?\\  specifies  \
```

Here are some examples:

```
(setq edt-word-entities '(?\t ?- ?/)) ;; Specifies TAB, - , and /
(setq edt-word-entities '(?\t)        ;; Specifies TAB, the default
```

You can also specify characters by their decimal ascii values:

```
(setq edt-word-entities '(9 45 47))   ;; Specifies TAB, - , and /
```

ENABLING EDT CONTROL KEY SEQUENCE BINDINGS:

Where EDT key bindings and GNU Emacs key bindings conflict, the default GNU Emacs key bindings are retained by default. Some diehard EDT users may not like this. So, if the variable edt-use-EDT-control-key-bindings is set to true in a user's .emacs file, then the default EDT Emulation mode will enable most of the original EDT control key sequence bindings. If you wish to do this, add the following line to your .emacs file:

```
(setq edt-use-EDT-control-key-bindings t)
```

SETTING SCROLL MARGINS:

Scroll margins at the top and bottom of the window are now supported. (The design was copied from tpu-extras.el.) By default, this feature is enabled with the top margin set to 10% of the window and the bottom margin set to 15% of the window. To change these settings, you can invoke the function edt-set-scroll-margins in your .emacs file. For example, the following line

```
          (edt-set-scroll-margins "20%" "25%")

sets the top margin to 20% of the window and the bottom margin to 25% of the
window.  To disable this feature, set each margin to 0%.  You can also invoke
edt-set-scroll-margins interactively while EDT Emulation is active to change
the settings for that session.

NOTE: Another way to set the scroll margins is to use the Emacs customization
feature (not available in Emacs 19) to set the following two variables
directly:

          edt-top-scroll-margin and edt-bottom-scroll-margin

Enter the Emacs 'customize' command.  First select the Editing group and then
select the Emulations group.  Finally, select the Edt group and follow the
directions.
```

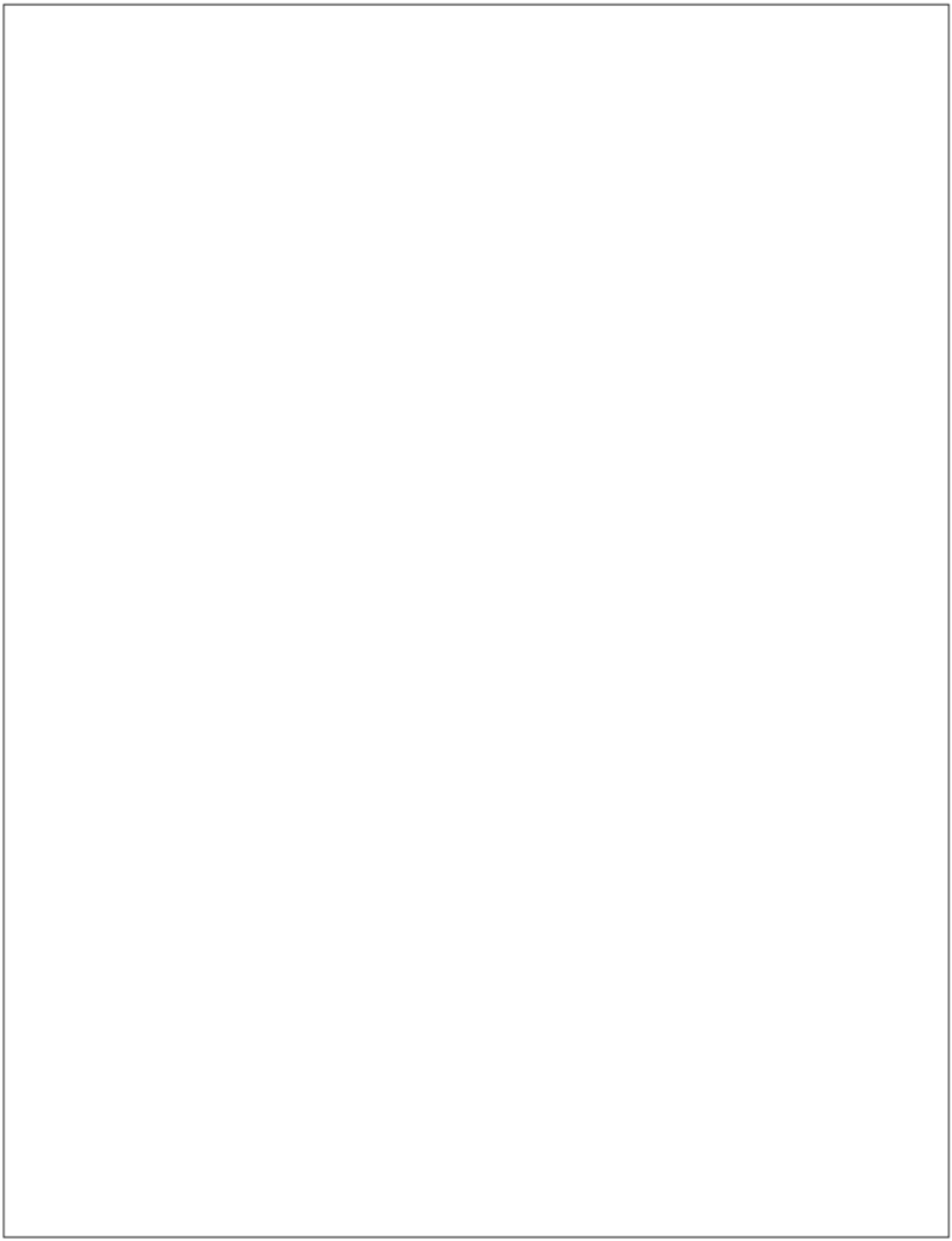
DEFAULT EDT Keypad

F7: Copy Rectangle
F8: Cut Rect Overstrike
G-F8: Paste Rect Overstrike
F9: Cut Rect Insert
G-F9: Paste Rect Insert
F10: Cut Rectangle
G-F10: Paste Rectangle
F11: ESC
F12: Beginning of Line
G-F12: Delete Other Windows
F13: Delete to Begin of Word
HELP: Keypad Help
G-HELP: Emacs Help
DO: Execute extended command
C-g: Keyboard Quit
G-C-g: Keyboard Quit
C-h: Beginning of Line
G-C-h: Emacs Help
C-i: Tab Insert
C-j: Delete to Begin of Word
C-k: Define Key
G-C-k: Restore Key
C-l: Form Feed Insert
C-n: Set Screen Width 80
C-r: Isearch Backward
C-s: Isearch Forward
C-t: Display the Time
C-u: Delete to Begin of Line
C-v: Redraw Display
C-w: Set Screen Width 132
C-z: Suspend Emacs
G-C-\: Split Window

G-b: Buffer Menu
G-c: Compile
G-d: Delete Window
G-e: Exit
G-f: Find File
G-g: Find File Other Window
G-h: Keypad Help
G-i: Insert File
G-k: Toggle Capitalization Word
G-l: Lowercase Word or Region
G-m: Save Some Buffers
G-n: Next Error
G-o: Switch to Next Window
G-q: Quit
G-r: Revert File
G-s: Save Buffer
G-u: Uppercase Word or Region
G-v: Find File Other Window
G-w: Write file
G-y: EDT Emulation OFF
G-z: Switch to User EDT Key Bindings
G-1: Delete Other Windows
G-2: Split Window
G-%: Go to Percentage
G- : Undo (GOLD Spacebar)
G-=: Go to Line
G-`: What line
G-/: Query-Replace

Prev Line (UP) Window Top	Next Line (DOWN) Window Bot	Bkwd Char (LEFT) Bkwd Sent	Frwd Char (RIGHT) Frwd Sent
GOLD (PF1) Mark Wisel	HELP (PF2) Desc Funct	FNDNXT (PF3) FIND	DEL L (PF4) UND L
PAGE (7) Ex Ext Cmd	SECT (8) Fill Regio	APPEND (9) REPLACE	DEL W (-) UND W
ADVANCE (4) BOTTOM	BACKUP (5) TOP	CUT (6) Yank	DEL C (,) UND C
WORD (1) CHNGCASE	EOL (2) DEL EOL	CHAR (3) Quoted Ins	Next Window (ENTER)
LINE (0) Open Line		SELECT (.) RESET	Query Replace

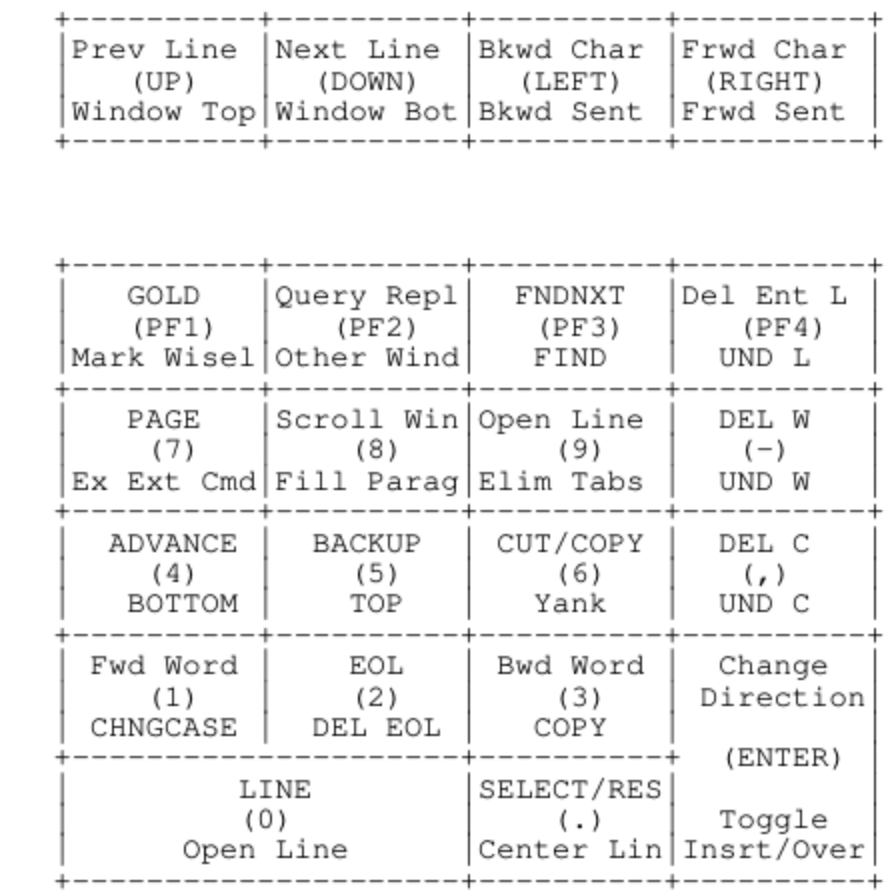
FNDNXT (FIND) FIND	Yank (INSERT)	CUT (REMOVE) COPY
SELECT/RES (SELECT)	SECT BACKW (PREVIOUS)	SECT FORWA (NEXT)

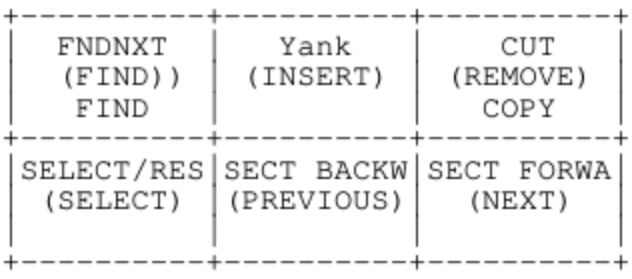


```
;;; File:  edt-user.el   ---  Sample User Customizations for the Enhanced
;;;                             EDT Keypad Mode Emulation
;;;
;;;                       For GNU Emacs 19 and Above
;;;
;; Copyright (C) 1986, 1992, 1993, 2000, 2001, 2002, 2003, 2004, 2005,
;;   2006, 2007  Free Software Foundation, Inc.

;; Author: Kevin Gallagher <[redacted]>
;; Maintainer: Kevin Gallagher <[redacted]>
;; Keywords: emulations

;; GNU Emacs is free software; you can redistribute it and/or modify
;; it under the terms of the GNU General Public License as published by
;; the Free Software Foundation; either version 2, or (at your option)
;; any later version.

;; GNU Emacs is distributed in the hope that it will be useful,
;; but WITHOUT ANY WARRANTY; without even the implied warranty of
;; MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
;; GNU General Public License for more details.

;; You should have received a copy of the GNU General Public License
;; along with GNU Emacs; see the file COPYING.  If not, write to the
;; Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor,
;; Boston, MA 02110-1301, USA.

;;; Commentary:

;; This file contains GNU Emacs User Custom EDT bindings and functions.

;;; Usage:

;;  See edt-user.doc in the emacs etc directory.

;; ======================================================================
```

```
;;;;
;;;; Setup user custom EDT key bindings.
;;;;

(defun edt-setup-user-bindings ()
  "Assigns user custom EDT Emulation keyboard bindings."

  ;; PF1 (GOLD), PF2, PF3, PF4
  ;;
  ;; This file MUST contain a binding of PF1 to edt-user-gold-map.  So
  ;; DON'T CHANGE OR DELETE THE REGULAR KEY BINDING OF PF1 BELOW!
  ;; (However, you may change the GOLD-PF1 binding, if you wish.)
  (edt-bind-function-key "PF1" 'edt-user-gold-map 'edt-mark-section-wisely)
  (edt-bind-function-key "PF2" 'query-replace 'other-window)
  (edt-bind-function-key "PF4" 'edt-delete-entire-line 'edt-undelete-line)

  ;; EDT Keypad Keys
  (edt-bind-function-key "KP1" 'edt-word-forward 'edt-change-case)
  (edt-bind-function-key "KP3" 'edt-word-backward 'edt-copy)
  (edt-bind-function-key "KP6" 'edt-cut-or-copy 'yank)
  (edt-bind-function-key "KP8" 'edt-scroll-window 'fill-paragraph)
  (edt-bind-function-key "KP9" 'open-line 'edt-eliminate-all-tabs)
  (edt-bind-function-key "KPP"
                         'edt-toggle-select 'edt-line-to-middle-of-window)
  (edt-bind-function-key "KPE" 'edt-change-direction 'overwrite-mode)

  ;; GOLD bindings for regular keys.
  (edt-bind-gold-key "a" 'edt-append)
  (edt-bind-gold-key "A" 'edt-append)
  (edt-bind-gold-key "h" 'edt-electric-user-keypad-help)
  (edt-bind-gold-key "H" 'edt-electric-user-keypad-help)

  ;; Control bindings for regular keys.
  ;;; Leave binding of C-c as original prefix key.
  (edt-bind-key "\C-j" 'edt-duplicate-word)
  (edt-bind-key "\C-k" 'edt-define-key)
  (edt-bind-gold-key  "\C-k" 'edt-restore-key)
  (edt-bind-key "\C-l" 'edt-learn)
  ;;; Leave binding of C-m to newline.
  (edt-bind-key "\C-n" 'edt-set-screen-width-80)
  (edt-bind-key "\C-o" 'open-line)
  (edt-bind-key "\C-p" 'fill-paragraph)
  ;;; Leave binding of C-r to isearch-backward.
  ;;; Leave binding of C-s to isearch-forward.
  (edt-bind-key "\C-t" 'edt-display-the-time)
  (edt-bind-key "\C-v" 'redraw-display)
  (edt-bind-key "\C-w" 'edt-set-screen-width-132)
  ;;; Leave binding of C-x as original prefix key.
)
```

```
;;;
;;; LK-201 KEYBOARD USER EDT KEYPAD HELP
;;;

(defun edt-user-keypad-help ()
  "
                                USER EDT Keypad Active
```






```
   F7: Copy Rectangle
   F8: Cut Rect Overstrike
 G-F8: Paste Rect Overstrike
   F9: Cut Rect Insert
 G-F9: Paste Rect Insert
  F10: Cut Rectangle
G-F10: Paste Rectangle
  F11: ESC
  F12: Beginning of Line
G-F12: Delete Other Windows
  F13: Delete to Begin of Word
 HELP: Keypad Help
G-HELP: Emacs Help
   DO: Execute extended command
  C-a: Beginning of Line
  C-b: Backward Character
  C-d: Delete Character
  C-e: End of Line
  C-f: Forward Character
  C-g: Keyboard Quit
G-C-g: Keyboard Quit
  C-h: Electric Emacs Help
G-C-h: Emacs Help
  C-i: Indent for Tab
  C-j: Duplicate Word
  C-k: Define Key
G-C-k: Restore Key
  C-l: Learn
  C-n: Set Screen Width 80
  C-o: Open Line
  C-p: Fill Paragraph
  C-q: Quoted Insert
  C-r: Isearch Backward
  C-s: Isearch Forward
  C-t: Display the Time
  C-u: Universal Argument
  C-v: Redraw Display
  C-w: Set Screen Width 132
  C-z: Suspend Emacs
G-C-\\: Split Window

  G-a: Append to Kill Buffer
  G-b: Buffer Menu
  G-c: Compile
  G-d: Delete Window
  G-e: Exit
  G-f: Find File
  G-g: Find File Other Window
  G-h: Keypad Help
  G-i: Insert File
  G-k: Toggle Capitalization Word
  G-l: Lowercase Word or Region
  G-m: Save Some Buffers
  G-n: Next Error
```

```
G-o: Switch Windows
G-q: Quit
G-r: Revert File
G-s: Save Buffer
G-u: Uppercase Word or Region
G-v: Find File Other Window
G-w: Write file
G-y: EDT Emulation OFF
G-z: Switch to Default EDT Key Bindings
G-2: Split Window
G-%: Go to Percentage
G- : Undo  (GOLD Spacebar)
G-=: Go to Line
G-`: What line
G-/: Query-Replace"

(interactive)
(describe-function 'edt-user-keypad-help))
```